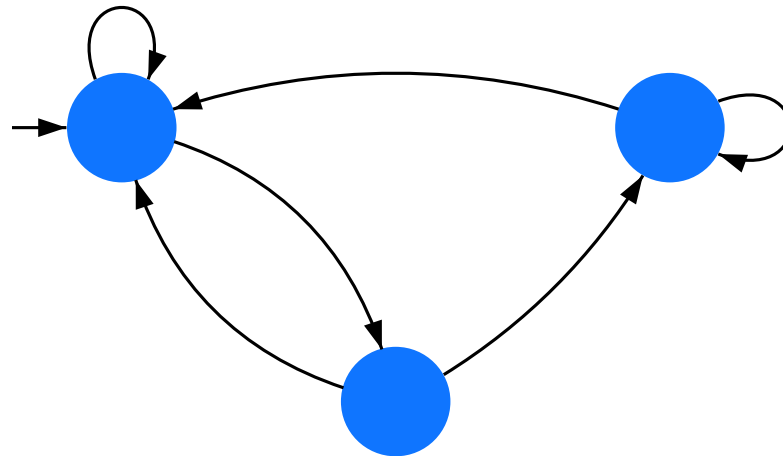


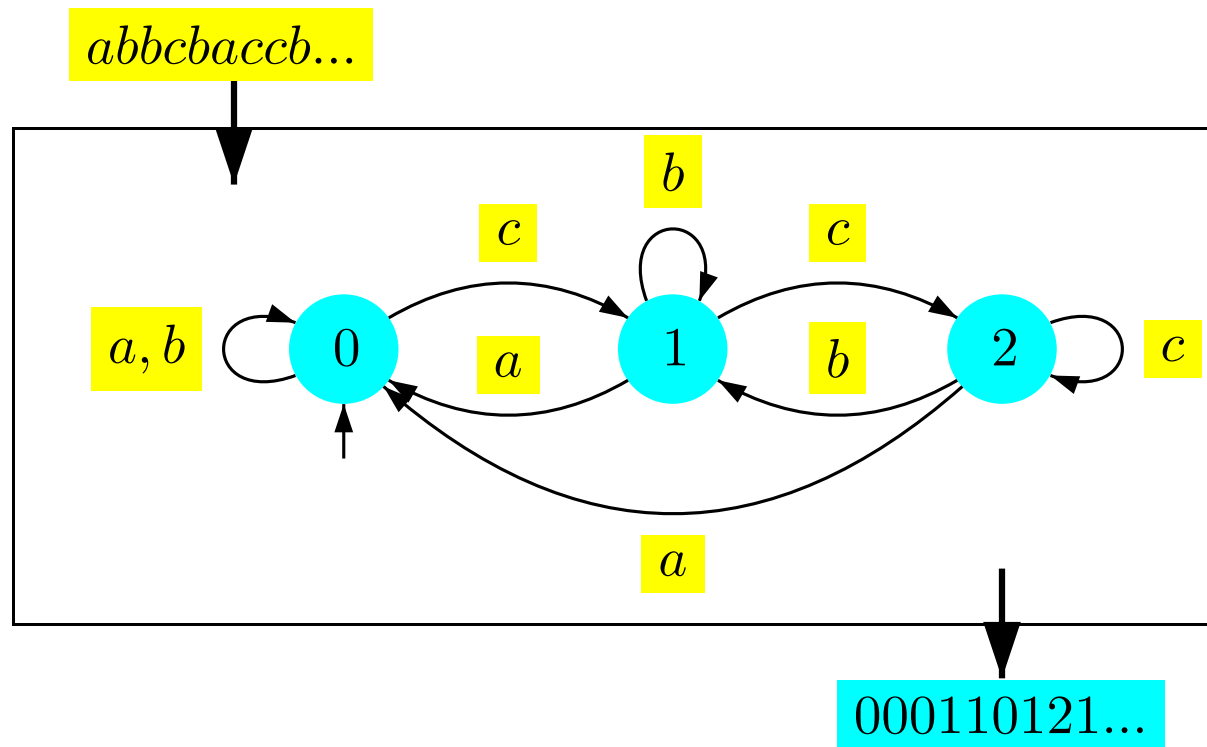
Vector Algorithms for Approximate String Matching ^a

Anne Bergeron et Sylvie Hamel



^aInternational Journal of Foundations of Computer Science, Vol.13, No.1, pp.53-65, 2002.

Ancienne Technologie: utilisation **séquentielle** d'un automate pour convertir une chaîne d'entrée en une chaîne de sortie.



Analyse en temps: $\mathcal{O}(m)$ où m est la longueur de la chaîne d'entrée

Vecteurs caractéristiques:

Étant donné la séquence d'entrée

$$e = abbcbaccb...,$$

on a les vecteurs caractéristiques suivants:

$$a = 100001000...$$

$$b = 011010001...$$

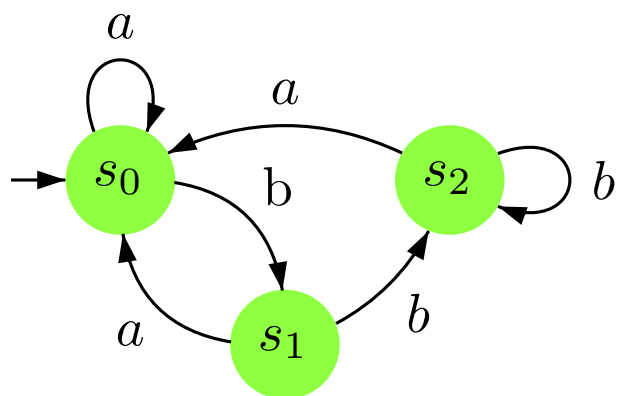
$$c = 000100110...$$

Opérations sur les vecteurs de bits

Soit x et y des vecteurs booléens - ou vecteurs de bits. On permet les opérations suivantes sur ces vecteurs:

- | | |
|----------------------------------|----------------|
| 1. disjonction | $x \vee y$ |
| 2. conjonction | $x \wedge y$ |
| 3. négation | $\neg x$ |
| 4. addition binaire avec retenue | $x \vdash_b y$ |
| 5. déplacement vers la droite | $\uparrow_a x$ |

Nouvelle Technologie: trouver la chaîne de sortie de façon **parallèle** en utilisant des **opérations vectorielles** pour calculer le vecteur caractéristique de chaque état.



		b	a	b	b	a
a	$=$	0	1	0	0	1
b	$=$	1	0	1	1	0

$$s_0 = a$$

$$s_1 = (\uparrow_1 s_0) \wedge b$$

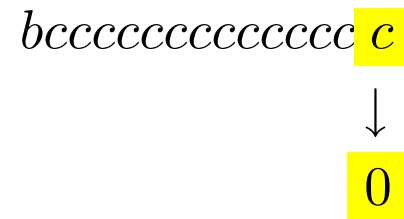
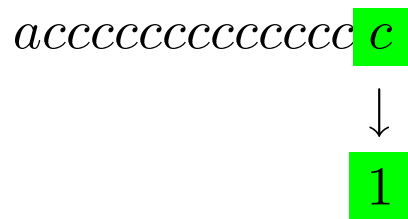
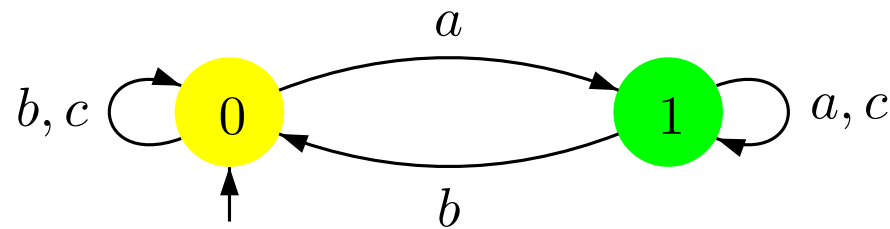
$$s_2 = \neg (s_0 \vee s_1)$$

$$s_0 = 0 \quad 1 \quad 0 \quad 0 \quad 1$$

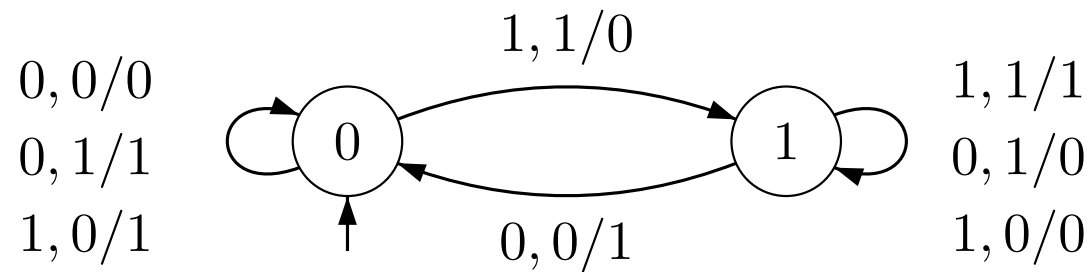
$$s_1 = 1 \quad 0 \quad 1 \quad 0 \quad 0$$

$$s_2 = 0 \quad 0 \quad 0 \quad 1 \quad 0$$

Se souvenir d'événements passés: une **sortie** peut dépendre d'un **événement** arbitrairement “éloigné”...



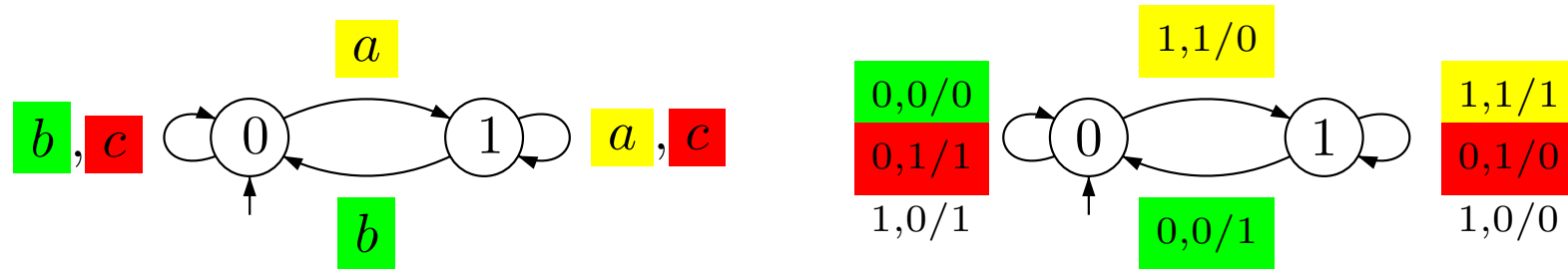
...mais il est encore possible de calculer, en parallèle, le vecteur caractéristique de chaque état en utilisant **l'automate d'addition binaire**:



Si deux vecteurs de bits $\mathbf{x}_1$ et $\mathbf{x}_2$ sont additionnés, la sortie sera 1 si

$$(\mathbf{x}_1 \wedge \mathbf{x}_2) \vee [((\neg \mathbf{x}_1 \wedge \mathbf{x}_2) \vee (\mathbf{x}_1 \wedge \neg \mathbf{x}_2)) \wedge \neg(\mathbf{x}_1 +_b \mathbf{x}_2)]$$

Maintenant, définissons $x_1 = a$ et $x_2 = a \vee c$:



Alors:

$a = x_1 \wedge x_2 =$ les deux bits sont 1

$b = \neg x_1 \wedge \neg x_2 =$ les deux bits sont 0

$c = \neg x_1 \wedge x_2 =$ le premier bit est 0 et le second est 1

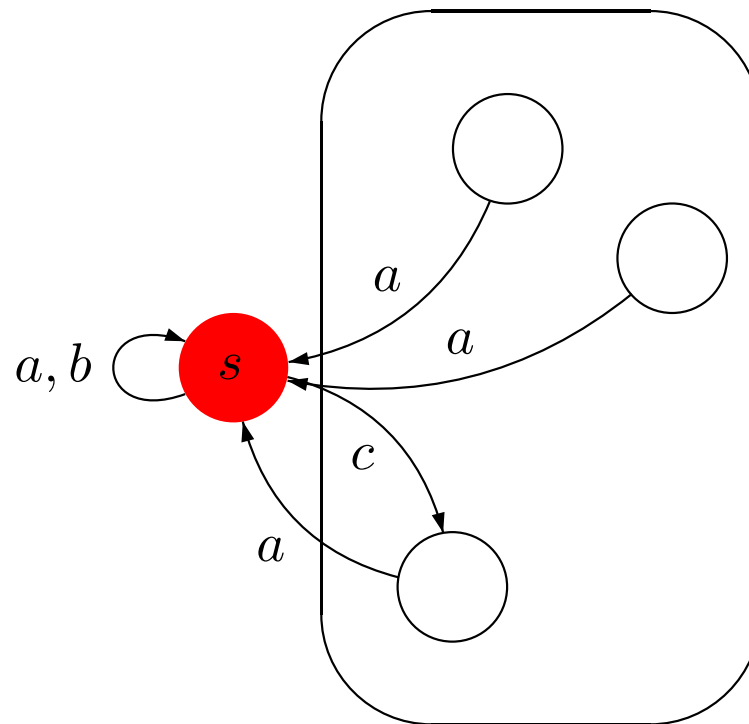
On obtient:

$$(r = 1) = a \vee [c \wedge \neg(a +_b (a \vee c))]$$

De façon équivalente, on trouve

$$(r = 0) = b \vee [c \wedge (\neg b +_b \neg(b \vee c))]$$

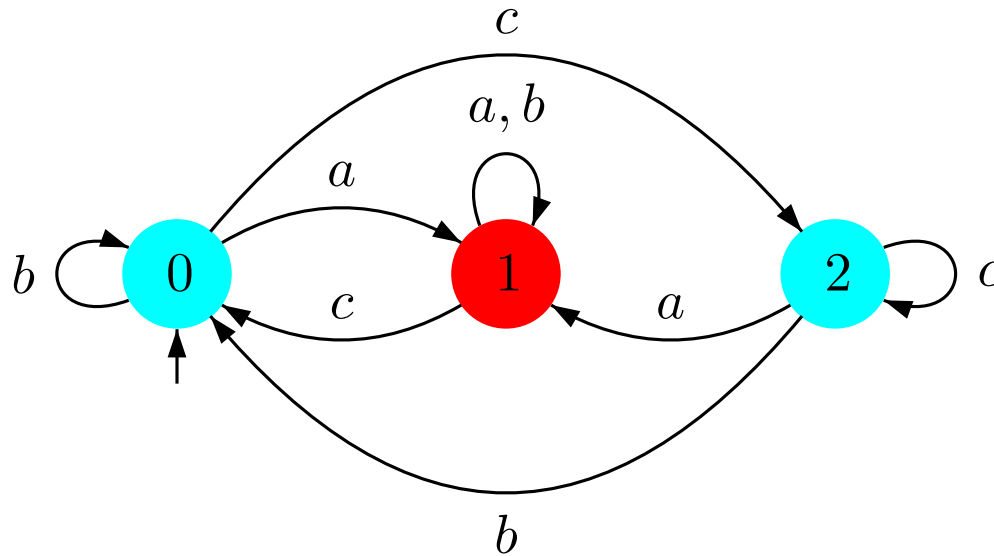
Résoudre des automates plus complexes: quels **automates** pouvons-nous **résoudre** en utilisant **nos techniques**?



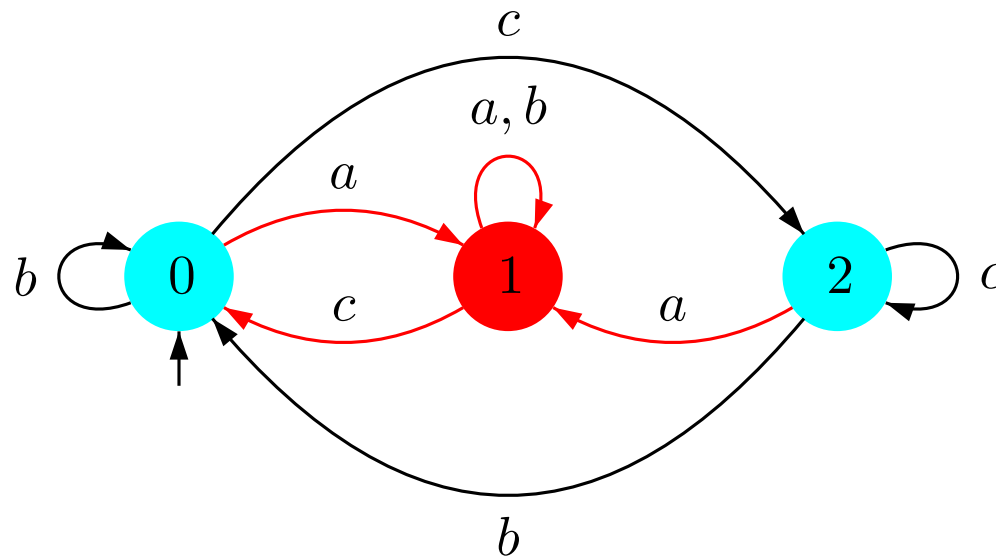
Quelques Définitions: soit $\mathcal{A}$ un automate complet déterministe sur l'alphabet Σ avec ensemble d'états Q et fonction de transition $Q \times \Sigma \xrightarrow{\delta} Q$.

Définition 1. Un état s est **effeuillable** si, pour tout événement $x \in \Sigma$, on a

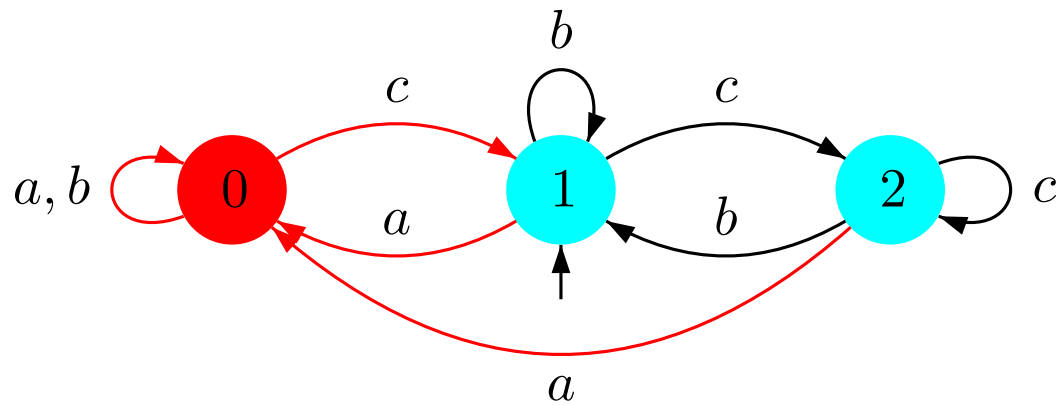
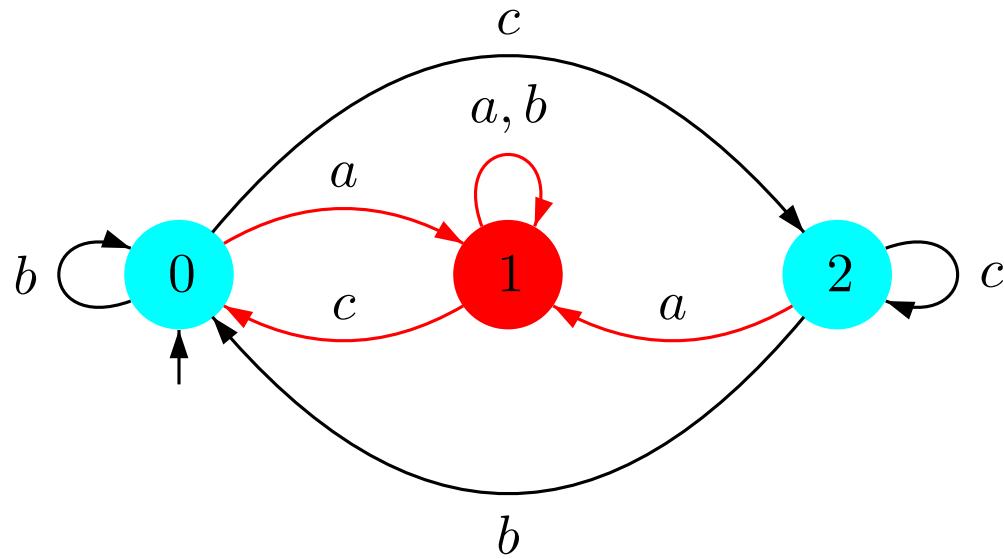
$$[\exists p \neq s \text{ tel que } \delta(p, x) = s] \Rightarrow [\forall q \in Q, \delta(q, x) = s].$$



Definition 2. Un automate $\mathcal{A}$ est *effeuillable* s'il a un seul état, ou s'il a un état effeuillable s et que $\mathcal{A} \setminus \{s\}$ est effeuillable.



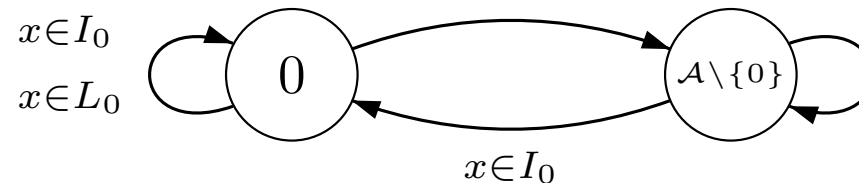
Ordre d'effeuillement:



Théorème 1. Si un automate $\mathcal{A}$ est effeuillable, alors on peut trouver un algorithme vectoriel pour $\mathcal{A}$.

Idée de la preuve:

1. Ordonner les d états de l'automate $\mathcal{A}$ par ordre d'effeuillement et supposer que l'ensemble d'états est $Q = \{0, \dots, d-1\}$.
2. On a alors la situation suivante:



et le lemme de l'addition implique que

$$(r = 0) = \begin{cases} I_0 \vee [L_0 \wedge (\neg I_0 + \neg(I_0 \vee L_0))] \\ I_0 \vee [L_0 \wedge \neg(I_0 + (I_0 \vee L_0))] \end{cases} \quad \text{ou}$$

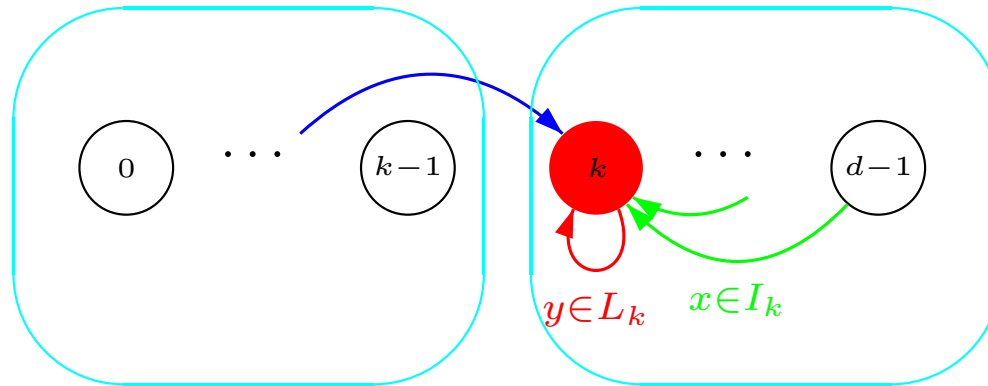
selon le fait que 0 soit l'état initial ou non.

3.

Idée de la preuve (suite):

3. Calcul de $(r = k)$ pour $k \in \{1, \dots, d-2\}$:

On connaît $K = (r < k)$ et on veut trouver $(r = k)$:



$$N \leftarrow [(\uparrow_i K) \wedge (\sigma(\uparrow_i r, e) = k)] \vee [\neg K \wedge (e \in I_k)]$$

$$(r = k) \leftarrow \begin{cases} N \vee [L_k \wedge (\neg N + \neg(N \vee L_k))] & \text{ou} \\ N \vee [L_k \wedge \neg(N + (N \vee L_k))] \end{cases}$$

$$K \leftarrow K \wedge (r = k)$$

4.

Idée de la preuve (suite):

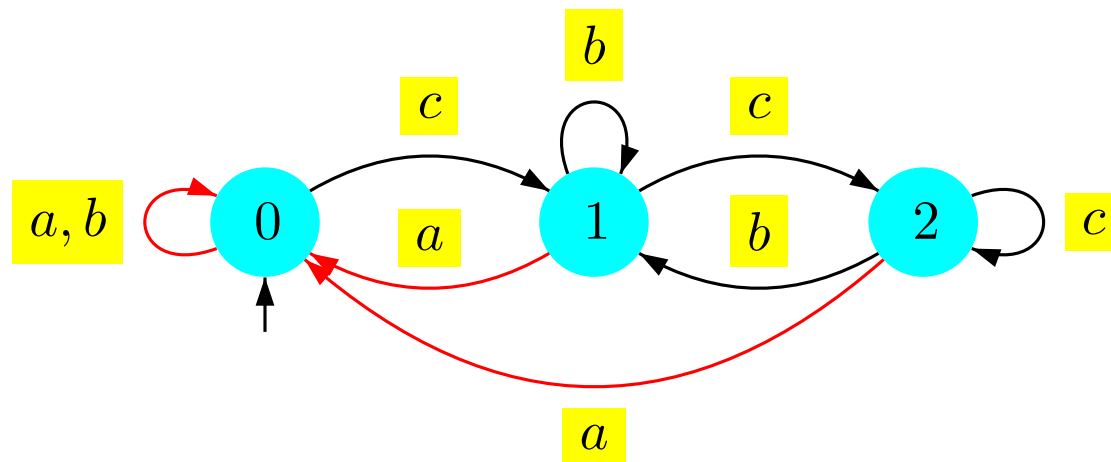
4. On calcule le vecteur caractéristique du dernier état:

$$(r = d - 1) = \neg K$$

Complexité: $\mathcal{O}(d|\Sigma|)$, où d est le nombre d'états de l'automate.

Exemple:

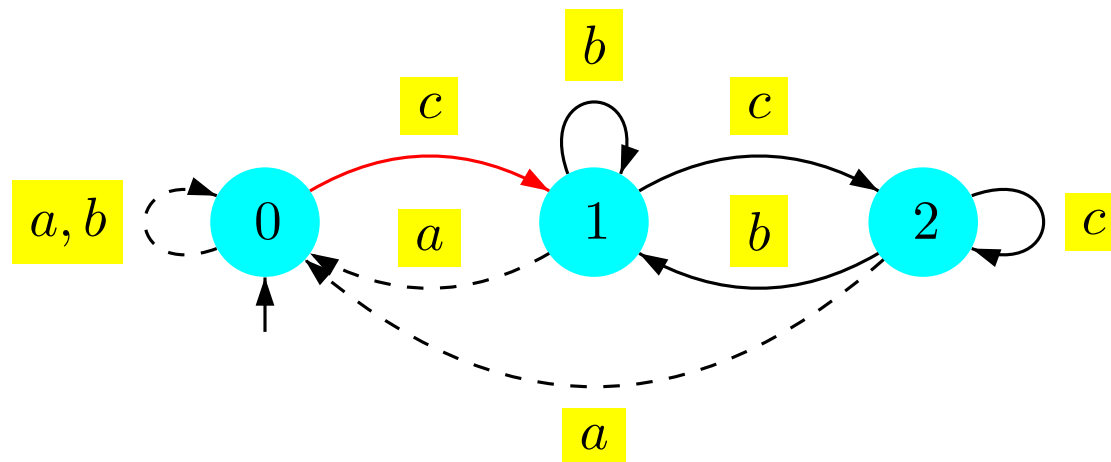
Étape 1. **Premier état effeuillable: 0**



a	b	b	c	b	a	c	c	b
0	0	0			0			

Exemple:

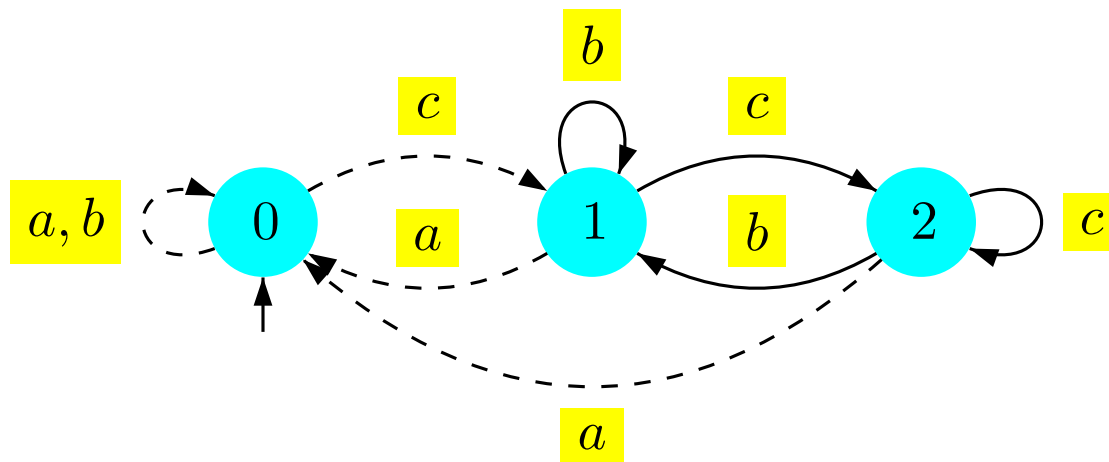
Étape 2. “Voisins de 0”



<i>a</i>	<i>b</i>	<i>b</i>	<i>c</i>	<i>b</i>	<i>a</i>	<i>c</i>	<i>c</i>	<i>b</i>
0	0	0	1		0	1		

Exemple:

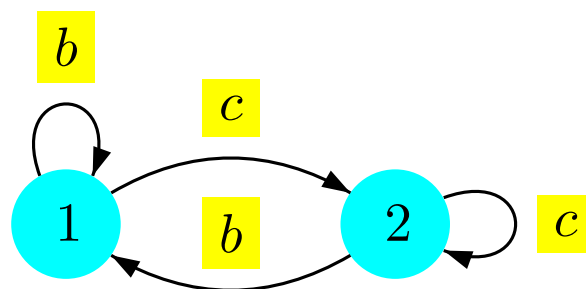
Étape 3. “Étape réursive”



<i>a</i>	<i>b</i>	<i>b</i>	<i>c</i>	<i>b</i>	<i>a</i>	<i>c</i>	<i>c</i>	<i>b</i>
0	0	0	1		0	1		

Exemple:

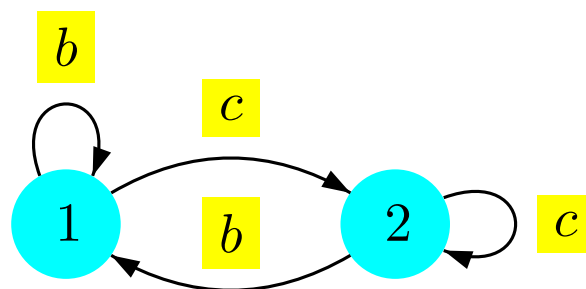
Étape 3. “Étape réursive”



<i>a</i>	<i>b</i>	<i>b</i>	<i>c</i>	<i>b</i>	<i>a</i>	<i>c</i>	<i>c</i>	<i>b</i>
0	0	0	1		0	1		

Exemple:

Étape 3. “Étape réursive”



<i>a</i>	<i>b</i>	<i>b</i>	<i>c</i>	<i>b</i>	<i>a</i>	<i>c</i>	<i>c</i>	<i>b</i>
0	0	0	1	1	0	1	2	1

Solution Classique: Calculer la matrice des distances $D[i, j]$

		<i>A</i>	<i>C</i>	<i>G</i>	<i>T</i>	<i>A</i>	<i>A</i>	<i>T</i>	<i>A</i>	<i>G</i>	<i>C</i>	...
<i>T</i>	0	0	0	0	0	0	0	0	0	0	0	...
<i>A</i>	1	1	1	1	0	1	1	0	1	1	1	...
<i>T</i>	2	1	2	2	1	0	1	1	0	1	...	...
<i>A</i>	3	2	2	3	2	1	1	1	1	1	...	...
<i>A</i>	4	3	3	3	3	2	1	2	1	2	...	...

↑

↑

où

$$D[i, j] = \min \begin{cases} D[i - 1, j - 1] + \delta(p_i, t_j) \\ D[i, j - 1] + 1 \\ D[i - 1, j] + 1 \end{cases} \quad (1)$$

avec les conditions initiales $D[0, j] = 0$ et $D[i, 0] = i$.

Nouvelle Solution: Calculer les distances avec un automate ...

Définition 3. On définit les différences horizontales et verticales entre les éléments de la table des distances D de la façon suivante:

$$\Delta v_{i,j} = D[i, j] - D[i - 1, j],$$

$$\Delta h_{i,j} = D[i, j] - D[i, j - 1].$$

Lemme 1. Les différences horizontales et verticales sont bornées, c'est-à-dire

$$\Delta v_{i,j} \in \{-1, 0, 1\},$$

$$\Delta h_{i,j} \in \{-1, 0, 1\}.$$

Connaître les différences horizontales nous permet de résoudre le problème...

		<i>A</i>	<i>C</i>	<i>G</i>	<i>T</i>	<i>A</i>	<i>A</i>	<i>T</i>	<i>A</i>	<i>G</i>	<i>C</i>	...	
		0	0	0	0	0	0	0	0	0	0	0	...
<i>T</i>	¹ 1	*										...	
<i>A</i>	¹ 2	*										...	
<i>T</i>	¹ 3	*										...	
<i>A</i>	¹ 4	*										...	

car on sait que $D[m, 0] = m$ et alors

$$D[m, j] = D[m, j - 1] + \Delta h_{m,j}.$$

Notre algorithme:

Au départ, on a que $\Delta \mathbf{v}_0 = 11 \dots 1$.

Pour j de 1 à n faire

1. Calculer $\Delta \mathbf{h}_j$ avec un automate $\mathcal{B}$.
2. Calculer $\Delta \mathbf{v}_j$:

$$\Delta \mathbf{v}_j = \Delta \mathbf{h}_j + \Delta \mathbf{v}_{j-1} - \uparrow_0 \Delta \mathbf{h}_j$$

3. Calculer $D[m, j]$

$$D[m, j] = D[m, j - 1] + \Delta h_{m, j}$$

Finalement, si $D[m, j] \leq t$ alors on a une occurrence de P en position j du texte.

$$D[i,j]=\min \begin{cases} D[i-1,j-1]+\delta(p_i,t_j) \\ D[i,j-1]+1 \\ D[i-1,j]+1 \end{cases} \quad \Rightarrow \quad \Delta h_{i,j}=\min \begin{cases} \delta(p_i,t_j)-\Delta v_{i,j-1} \\ \Delta h_{i-1,j}-\Delta v_{i,j-1}+1 \\ 1 \end{cases}$$

On peut donc définir un automate $\mathcal{B}$ qui calculera la colonne de différences horizontales $\Delta \mathbf{h}_j = \Delta h_{1,j} \dots \Delta h_{m,j}$ étant donné la séquence de couples suivante:

$$(\Delta \mathbf{v}_{j-1}, \delta(\mathbf{P}, \mathbf{t}_j)) = ((\Delta v_{1,j-1}, \delta(p_1, t_j)), \dots, (\Delta v_{m,j-1}, \delta(p_m, t_j)))$$

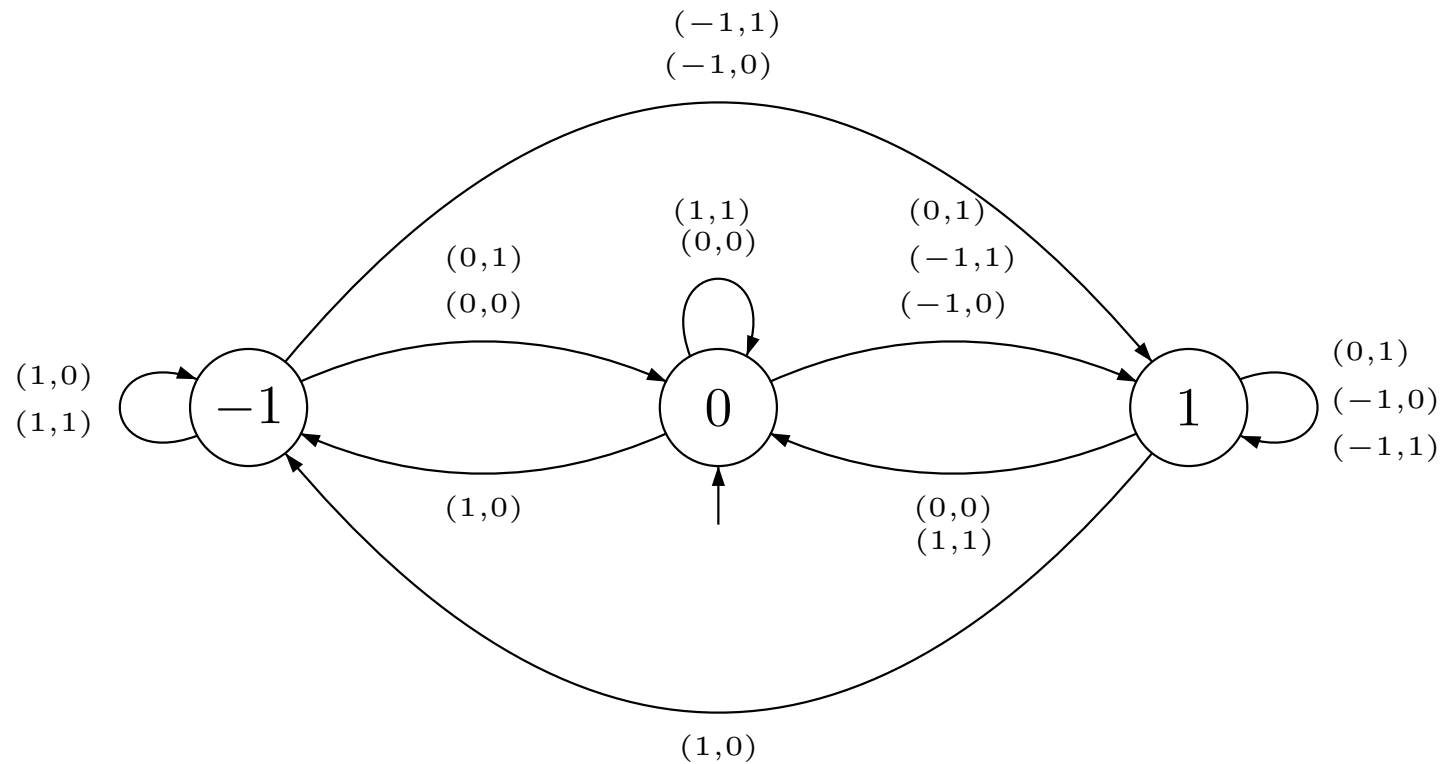
Les états de l'automate $\mathcal{B}$ sont $\{-1, 0, 1\}$, l'état initial est 0, et la fonction de transition σ de $\mathcal{B}$ est donné par:

$$\Delta h_{i,j} = \min \begin{cases} \delta(p_i, t_j) - \Delta v_{i,j-1} \\ \Delta h_{i-1,j} - \Delta v_{i,j-1} + 1 \\ 1 \end{cases} \implies \sigma(s, (\Delta v, \delta)) = \min \begin{cases} \delta - \Delta v \\ s - \Delta v + 1 \\ 1 \end{cases}$$

pour un événement $(\Delta v, \delta)$ dans le produit cartésien $\{-1, 0, 1\} \times \{0, 1\}$. On obtient la table de transitions suivante:

	(-1,0)	(-1,1)	(0,0)	(0,1)	(1,0)	(1,1)
-1	1	1	0	0	-1	-1
0	1	1	0	1	-1	0
1	1	1	0	1	-1	0

L'automate $\mathcal{B}$ permettant de calculer les colonnes de différences horizontales dans le cas où la distance utilisée est la distance d'édition est donc le suivant:



Théorème (Myers 99) ^a. L'automate à 3 états permettant de résoudre le problème de la recherche approximative d'un mot dans un texte, avec l'emploi de la **distance d'édition**, est effeuillable.

^aG. Myers, A Fast Bit-Vector Algorithm for Approximate String Matching Based on Dynamic Programming, Journal of the ACM, 46-3, pp. 395-415, 1999.

Theorem 2. L'automate $\mathcal{B}$ à $2c + 2$ états permettant de résoudre le problème de la recherche approximative d'un mot dans un texte, avec l'emploi de la **distance d'édition généralisée**, est effeuillable.